e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

STOCK PRICE PREDICTION USING AN ATTENTION- BASED LSTM HYBRID MODEL

Seera Naveen SaiKumar[*1]
[*1]Student, Computer Science, GMRIT, India.

ABSTRACT

Forecasting the stock price has been one of the grand challenges in the financial sector. With the discovery of machine learning models, especially deep learning architectures, research and innovation in financial forecasting have been nothing less than revolutionary. Over recent years, one observes that transformer architectures originally conceived for natural language processing tasks has emerged as a promising approach to time series prediction, including stock price forecasting. This paper covers the use of transformer. The model for the stock price forecast, with the self-attention mechanism, is applied because it can implement complex dependences and relations in the domain of financial time series. Indeed, it turns out that transformers model long-range dependencies much more efficiently than traditional recurrent neural networks (RNNs) or long short-term memory (LSTM) networks tends to have a problem with vanishing gradients .The method proposed is optimization of the accuracy of prediction by doing an all-rounded study on raw data from stocks regarding the price movement inclusive of the following: Historical prices, Technical indicators ,External market factors utilizing preprocessing techniques such as normalization and feature engineering, optimizing input into the transformer model testing has been done using traditional machine learning models and LSTM-based architectures, and it's proven to demonstrate that transformers are much better at capturing intricate temporal patterns as well as trends in the data than the latter two architectures. Results demonstrate that a transformer-based model goes better than traditional models in terms of both accuracy and computational efficiency. The paper was concluded by discussing possible implications of using transformers in financial forecasting, real-world implementation challenges such as noisy data noise, and market volatility

Keywords: Stock Price Prediction, Transformers, Time Series Forecasting, financial Forecasting, Self- Attention Mechanism, Deep Learning, Machine Learning, LSTM, Feature Engineering, Market Analysis.

I. INTRODUCTION

Stock price prediction had always been of great importance to financial and academic circles because it held the prospect of great economic gains and excellent opportunities for strategic investment. Even the efficient market hypothesis that proposes that stock prices cannot be predicted because they are inherently volatile and take immediately huge amounts of information into themselves has been proved by research that if such a predictive model is well-crafted and optimized, the results could be very accurate. in forecasting stock movements. These models depend mainly on the chosen input feature, the algorithm used, and the optimization techniques adopted for the model. The two historical techniques of forecasting stock prices have been through statistical tools using ARIMA models and economic tools like Granger causality analysis. These techniques have proved very successful in cases where most movements are determined by Trends and seasonality; however, their performance deteriorates drastically when applied to highly volatile or noisy finanial time series data. In recent years, the field has trended towards machine learning and deep learning techniques, especially CNNs and LSTM networks. The respective methods have been shown to effectively discover complex, non-linear relationships in time-series data of finances by exploitation of patterns in historical data to help improve the predictive power of predictions. Despite these exciting breakthroughs brought by deep learning methods, there are also some substantial challenges, including the fact that stock price movements can be caused by vast collections of the external, latent and interrelated factors. Although LSTMs have achieved impressive recognition in modeling sequential data, they oftentimes have problems with capturing long term dependencies and complex temporal relationships. This is where the transformer model comes out as a transformative architecture. Originally developed for natural language processing, transformers have gained much momentum for their efficiency in processing sequential data by doing away with the conventional bounds of traditional recurrent neural networks. Their self-attention mechanism allows for a global understanding of input data, making the model capture complex dependencies extended across time effectively. This paper envisages verification of advanced deep learning models, especially transformer. The architectures are tested on historical stock index data and individual stock prices. This robust metrics coupled with the walk-forward validation strategy ensures the model adapts to changing market conditions in a way that it simulates real-world scenarios as close as possible. We aim to prove through this methodology that proper architectural design along with right parameter tuning indeed significantly improves stock price prediction accuracy.

II. RELATED WORK

Numerous studies have explored various deep learning models for stock price prediction, each highlighting unique strengths and limitations that contribute to the field's advancement [1]. Wen and Li (2023) proposed an innovative LSTM-Attention- LSTM model, which effectively integrates the sequential modeling power of LSTM networks with the attention mechanism to enhance time-series prediction accuracy. This model improves focus on significant time steps, enabling better handling of intricate data patterns and boosting overall forecasting precision [2].Moghar and Hamiche (2020) demonstrated the efficacy of LSTM networks in capturing complex temporal dependencies present in financial data, showcasing their superiority over traditional statistical approaches such as ARIMA models. Their study highlighted that while LSTMs can outperform conventional methods, they face challenges in managing non-linear relationships and data with limited availability [3].Fan et al. (2024) introduced xLSTM models that address some of the limitations found in traditional LSTMs. By improving long-term forecasting capabilities and handling extended sequences more efficiently, xLSTM models reduce training parameters and optimize resource use, making them suitable for large-scale financial data applications. Bharathi and Geetha (2017) investigated the integration of sentiment analysis with historical stock data, illustrating a positive correlation between public sentiment derived from news and social media and market trends. However, they noted the constraints imposed by limited data diversity, which could affect the generalizability of their results [5].Venkataramana et al. (2024) proposed a hybrid model that combines LSTM networks with sentiment analysis, aiming to leverage the strengths of both techniques to capture historical price patterns and market sentiment. This integration resulted in enhanced predictive performance, demonstrating the importance of external variables in refining stock market predictions

[6].Li, Zhang, and Zhu (2023) explored a model that combines LSTM networks with attention mechanisms, emphasizing the importance of focusing on key time points within historical data. This approach was shown to improve the accuracy of stock price forecasting by allowing the model to weigh the relevance of different data points dynamically [7].Mehtab and Sen (2020) utilized CNN-LSTM architectures to leverage the complementary strengths of convolutional neural networks (CNNs) and LSTM networks. While CNNs excel at capturing local patterns within data, the LSTM component is adept at modeling long- term dependencies, resulting in a powerful model that improves predictive capabilities for financial time series[8].Zhou (2024) integrated attention mechanisms with LSTM to create a hybrid model that focuses on relevant historical data for enhanced predictive performance. The addition of attention layers enables the model to better prioritize impactful sequences, contributing to more accurate stock price predictions [9].Zhang et al. (2023) proposed a sophisticated CNN-BiLSTM-attention model that combines multiple advanced techniques. CNNs are used for extracting local features, BiLSTM for understanding bidirectional dependencies, and attention mechanisms for emphasizing important data points. This comprehensive approach proved effective for stock market forecasting by capturing a broader range of temporal and feature-based relationships

[11].Qiu et al. (2020) employed a model that combines LSTM networks with attention mechanisms to focus on influential sequences within time-series data, enhancing the predictive precision of the system. The attention mechanism enables the model to allocate varying levels of importance to different time steps, resulting in more nuanced predictions [12].Pardeshi et al. (2023) introduced sequential self-attention into an LSTM framework, improving the model's ability to handle long-term dependencies in financial data. This enhancement allowed for better management of complex temporal dynamics, leading to more robust and accurate stock price forecasts [13].Sang and Li (2024) developed a variant of LSTM with integrated attention, which strategically focuses on essential data sequences to improve prediction accuracy. This model demonstrated significant potential in analyzing intricate market patterns and responding effectively to dynamic changes in stock prices [14].Behura et al. (2024) presented a multi-layered sequential LSTM model aimed at processing sequential data for financial forecasting. By layering LSTMs, the model extracts deeper patterns over time, enhancing its

capability to predict stock prices accurately [15].Hargreaves and Leran (2020) highlighted the use of LSTM networks for real-time stock price prediction. Their work underscored the challenges of computational efficiency when handling large datasets, emphasizing the need for optimized architectures to maintain performance without compromising speed or accuracy.

This overview illustrates the rapid evolution of stock price prediction models, with researchers continuously refining techniques to address existing challenges and achieve higher predictive accuracy. Integrating attention mechanisms, hybrid architectures, and external variables such as sentiment analysis has proven pivotal in pushing the boundaries of what these models can accomplish.

III. METHODOLOGY

The main goal of this study is to develop a robust framework for stock price prediction using transformer models. For training and testing, we use historical stock data, including attributes such as date, open, high, low, close, and volume, which will be preprocessed to ensure data quality and consistency.

3.1 DATA COLLECTION

The quality, variety, and relevance of the data used for training, validation, and testing determines the success of any given stock price prediction model. For the case of stock price prediction using transformer models, the collection process must be carefully curated to capture a wide range of historical market information as well as external factors that may impact the movement of stocks. In the approach described here, a detailed breakdown of the strategy followed in gathering the data:

Historical stock data: The major source of this data is historical price information of major stock indices and individual stocks, which can be obtained through financial data providers like Bloomberg, Yahoo Finance or databases of stock exchanges.

Trading metrics: The data should contain all opening, closing, high, and low price records coupled with trade volume and market capitalization data.

Economic Indicators: The addition of macroeconomic variables, such as GDP growth, unemployment rates, and inflation, can be used to gain deeper insights into the trends that persist in the market.

Sentiment Data: Incorporation of news articles, social media sentiment-for example, feeds from Twitter-and financial news sentiment analysis can be made to take into account public perception that affects the movement of stock price.

3.2 DATA PREPROCESSING:

Normalization: Features such as stock prices and trading volume should be normalized to a specific range, typically within 0 and 1, to make them uniform.

Missing Data Handling: Techniques such as forward fill or interpolation would be used to mitigate missing data points ensuring continuity as well as the reliability of datasets.

Feature Engineering Identify some relevant features that would be helpful to capture market trends and patterns Generate lagged features to represent history of price movements and momentum as well as Presence of temporal information like day of the week or seasonality.

3.3 MODEL DESIGN:

Our forecasting framework benefits from the transformer structure, which we adopt for its self-attention mechanism, likely to capture the long-range dependencies inherent in time series data. We also make use of positional encodings to ensure that the order of input data is correctly maintained. The architecture also comprises multi-head attention layers and feed-forward neural network components to learn complex relationships within the data.

IV. ARCHITECTURE

Transformer model consists of an encoder-decoder structure. The encoder transforms input information into representations that are sensitive to contextual relationships between the elements. Decoder accepts the output of the encoder and one element at a time produces final predictions, paying attention to previously generated outputs as well.

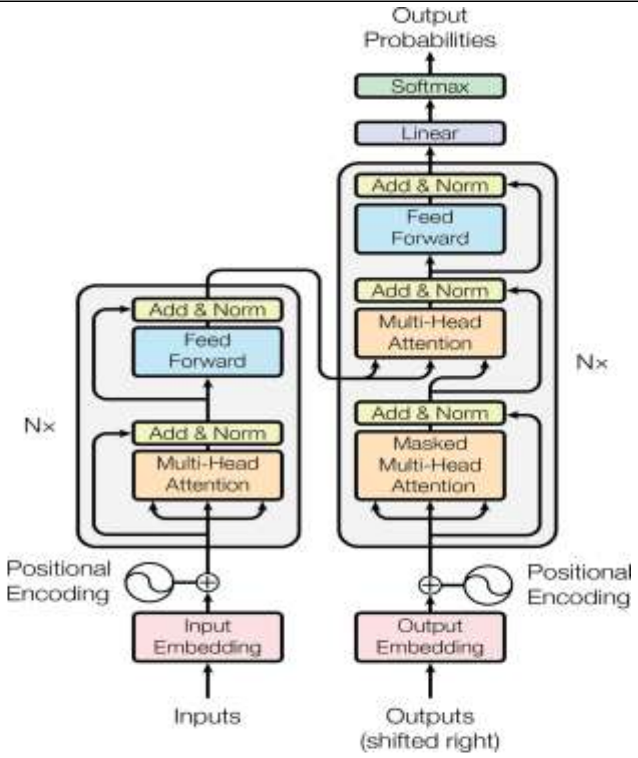

Fig 1: Architecture of transformers

(Source: https://images.app.goo.gl/ScnU8csLRMFXdS4B9)

Key modules:

Input Representation:

Tokenization: Input data-for example, text or sequential data-broken down into smaller units called tokens (words or subwords)-and fed into the model

Embedding: These tokens are converted to a dense vector representation, which the model can interpret numerically

Positional Encoding: Since the transformers don't understand the order in sequences, positional encodings are added to embeddings to keep a record of token positions in the sequence.

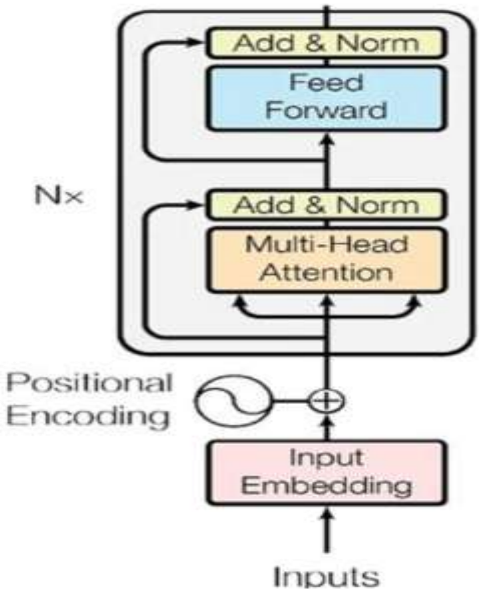

Fig 2: Positional encoding

(Source: https://images.app.goo.gl/ScnU8csLRMFXdS4B9)

Encoder Layers:

Self-Attention Mechanism: The core component that allows the model to focus on different parts of the input data simultaneously establishing how much attention each token should pay to every other token in the sequence so that the model captures the dependencies irrespective of the distance between them in the input it consists of three main components: Query (Q), Key (K), and Value (V) matrices.

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

(Ref: Vaswani, A. (2017). Attention is all you need. Advances in Neural Information Processing Systems.)

Feed-Forward Neural Network: That conducts a non-linear transformation separately over each representation for every token such that the model's ability to learn complex relations would be improved.

Residual Connections: Incorporate input from every sub-layer to its output for better gradient flow overcoming vanishing gradient problems in backpropagation.

Layer Normalization: Normalizes the output of every sub-layer for stable and faster training.

Decoder Layers:

Masked Self-Attention: Similar to the self-attention mechanism of the encoder however with this masking mechanism that prevents it from being able to "peek" ahead and try to predict the next token based on the future tokens. This is crucial for producing output in a sequence

Cross-Attention: It makes it so that the decoder can be more concerned about the right parts of the output that the encoder produces with each token as it generates the tokens in the output sequence.

Feed-Forward Neural Network and Residual Connections :It works same as in the encoder for processing the output.

Output:

Final Linear Layer: The processed output of the decoder transformed into a vector of size either for the target vocabulary or range of outputs.

Softmax Layer: Use softmax over all such possible tokens to get the probabilities for final predictions. Training Mechanisms:

Loss Function: Categorical cross-entropy will be used while evaluating the error between predictions and actual output.

Backpropagation: Update model's weights based on calculated gradients from loss to optimize the network's performance in iteration.

Attention Mechanism:

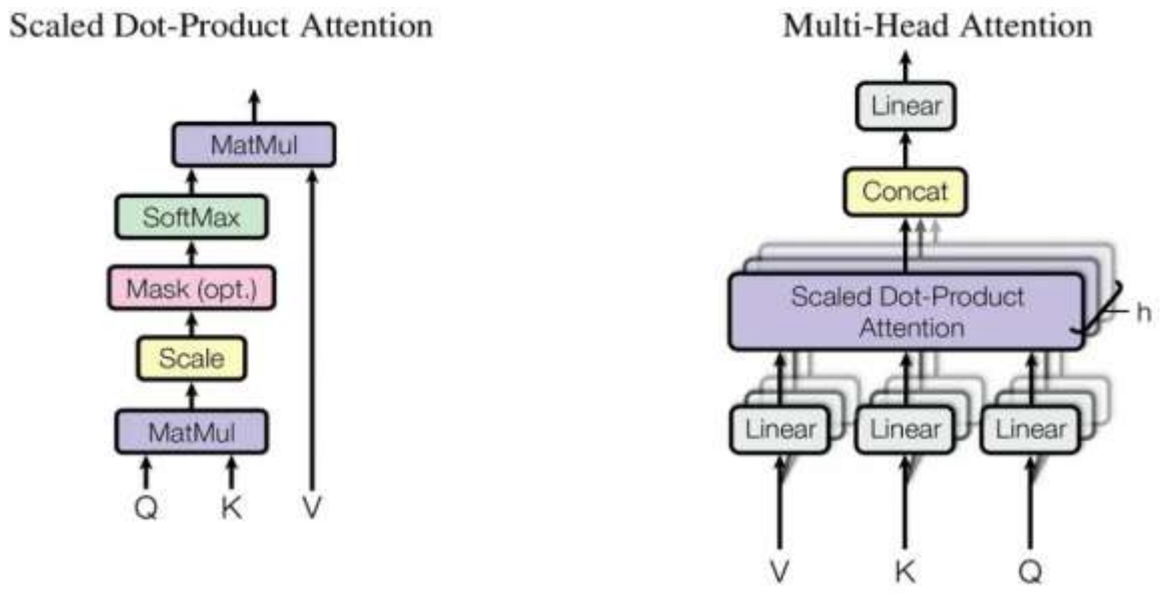

Fig 3: Attention mechanism

(source: https://vitalflux.com/wp-content/uploads/2024/01/attention-mechanism-in-transformer-1280x720.png)

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:06/Issue:11/November-2024 Impact Factor- 8.187 www.irjmets.com

Scaled-Dot Product Attention: Allows each token in the sequence to weigh the importance of other tokens when forming its representation. This enables the model to understand context and relationships between all parts of the input, such as the significance of certain words in a sentence or key time steps in time series data.

Multi-Head Attention: Improves the model's ability to focus on different aspects of the input by running multiple self-attention mechanisms in parallel and combining their outputs.

V. TRANSFORMERS WORK FLOW

1. Input Representation:

Data Preparation: The data we would most likely use in our application consists of daily stock metrics such as closing price, opening price, high and low prices, trading volume, and some other relevant variables like news sentiment, market indices, or macroeconomic indicators. Most of the mentioned data points are continuous and measure on different scales. So, we would have to normalize them, for example by Min-Max scaling, to have the same range across features and thus help the model learn better.

Embedding Layer: Since the transformer needs to feed into dense vectors, we need to transform our raw data into an appropriate form. To that end, we apply an embedding layer to the data of each day such that each day's data will be transformed into its vector form; for example, 60 data points since we have history for 60 days. In this embedding layer, each feature is represented as a dense vector. This makes this representation compress relevant information into a fixed size vector and, therefore, making the job easier for the transformer to process and still retain all the important features of the data originally presented.

2. Positional Encoding

This would mean that positional encoding injects information about the position in a sequence directly into the embeddings. For example, if we had because of this, if we wanted to use stock data over the last 60 days, we'd make different encodings for each of the first 60 days. These are sine functions including sine and cosine waves with different frequencies-and are arranged so that every position gets a unique encoding, as it also comes with the ability to compare encoding vectors at any position.

After the embedding of the data of each day's stock with the positional encoding for each input, we get a vector series where the information each vector contains information on the particular day's stock data as well as its position in the sequence.

3. Transformer Encoder:

At the core is an encoder, consisting of multiple layers, and the transformer models several self-attention mechanism followed by a feed-forward network that can be used for the observation of various levels of complexities that the model has learned while such as putting its data.

Self-Attention Mechanism:

The self-attention mechanism enables the model to focus on one or another part of the input sequence according to the task. for example in the case of stock price prediction, self-attention enables the model to dynamically weigh the data from different days' stock, putting greater significance to the recent data points if they are more predictive of tomorrow's price

Process:

1. Query, Key and Value Vectors: For each day in the input sequence, the transformer computes three vectors: Q, K and V. Q: The "question" of the day, meaning what information the model should look for in the data.

K: The data of other days that can potentially answer a query

V: The actual data we wish to extract once identified, which will be the relevant information.

2. Calculating Attention Scores: Each day's Query vector is compared with all other days' Key vectors by calculating the dot product, which gives a measure of relevance or attention score. A softmax function is then applied to normalize these scores, resulting in a set of attention weights for each day.
3. Weighted Sum of Values: The attention weights now calculate a weighted sum of Value vectors in such a way that the model draws focus to the Value vectors of the more relevant days and downplays the less relevant ones.

This way, the final output is a context-aware vector for every day that draws information from other days deemed relevant for the prediction to be made.

Multi-Head Attention:

Every head computes its own attention and then concatenates the vectors resulting from applying attention-weighted to all heads. The output with the concatenated result, due to its importance, is exposed to a linear transformation that manages to integrate all the different views into a representation of one only.

No. of multi-head attention modules: Prepare robust representations capturing the ability to draw multiple patterns and trends. Since these things are present in historical data, they might help in better forecasting.

4. Feed-Forward Neural Network:

It makes it more conducive to introduce non-linearity so that it is able to learn complex transformations and potentially higher it orders interactions among its features. Then, for each day, the feed-forward network generates an updated representation that captures

Those are more complex patterns than what attention mechanisms can individually produce

5. Residual Connections and Layer Normalization

Additional residual connections and layer normalization are further provided to transformers to simplify training.

Residual Connections: In this scenario, it is comprised of a skip connection; that is, it is comprised of the input directly to the output of any sub layer-for example, multi-head attention and feed-forward. Since the model can carry information in earlier layers by virtue of having no vanishing gradient problem because of the skip connection, the sublayers can be very deep.

Layer Normalization: The output from each layer like self-attention or feed-forward is normalized after every layer so that the process of training is stabilized and convergence is accelerated.

The output vectors are the input refinements from the original inputs, with information on local and global trends in the stock data:

6. Decoder and Prediction Mechanism

Input to the final layer, in general, is the fully connected layer that maps the output to a continuous space, is the encoder's output. It uses the weighted information of all previous days to produce a final output, creating thereby one single prediction for next day. Last Prediction of the stock price on day 61 based on the learned pattern and historical data

7. Training of the Model:

The model needs actual stock prices as targets that give the most accurate predictions. The training processes are done in multiple steps and thus can be very computationally expensive. The MSE, loss function is used as mean squared error since it is common to use while doing regression tasks, such as when dealing with stock price predictions, since it penalizes large errors of the actual stock prices more severely. MSE computes differences between actual and predicted stock prices, squares them all, and takes an average.

Back-propagation and Optimization: One calculates gradients of the loss function with respect to the model weights by back-propagation. The back-propagation computation of the gradients adjusts the weights in the direction that minimizes the loss. Most of the time, one uses an optimizer, such as Adam, to update the weights; the optimizer automatically adjusts the learning rate to minimize the convergence time.

Regularization and Dropout: Because of the non-deterministic behavior of stock prices, it could easily overfit and so regularization techniques, like dropout may be used in some of the layers. This randomly "drops" units during training. In this case, the model does not become over-dependent on certain features it has learned and hence will generalize better to unseen data.

8. Inference: Using the Trained Model for Prediction

This model, upon training, can be used to predict the stock prices moving into the future. In any prediction, it accepts the last 60 days' worth of historical data and feeds that into the encoder layers to produce a forecast. This architecture thus lets the model be used as a predictor in a recursive manner, predicting stocks several days into the future.

Summary of Data Flow Through the Transformer

Input: Preprocessed data in the form of stocks and its features, and the positional encoding.

Self-Attention and Multi-Head Attention: Learning data internal dependencies by giving prime importance to recent patterns, however acknowledging old ones too
Feed-Forward Network: Including non-linearity, representing more complex relationships within the data
Decoder/Prediction Layer: Output of the actual kind that consists of lastly forecasted price of the stock for the next day.

VI. CONCLUSION

In the case of stock price prediction, determining the observations and giving them the right weights would play a most significant role in the precision of the estimated value. Since the architecture of the transformer is very robust, based on self- attention mechanism, layered structure, and the non-linearity of feedforward networks, it tends to become an ideal building block for Such a task is developed. Depth to the model comes by subcomponents-embedding and positional encoding and multihead attention up to feed-forward layers to represent both short and long patterns in data.
For instance, If in a firm that transformer model utilized 60 days' history stock data in making a prediction for tomorrow's price. Training makes it realize that it has to use weights dynamically in self-attention so that it is not always focusing on different days as necessary in order to make a prediction. After several rounds of training and fine-tuning, the model starts making pretty accurate predictions based upon a holistic view of past price movements, trading volumes, and other economic factors. As the transformers improve, using methodologies that involve even reinforcement learning with transformers or hybrid models such as LSTMs, further greater capabilities are opened up to raising the accuracy of the predictions. However, for now, it is the transformer that brings the powerful tool for stock prediction, forming a basis for much more in the near future related to the advancements of deep learning.

VII. REFERENCES

[1] X. Wen and W. Li, "Time Series Prediction Based on LSTM-Attention-LSTM Model," in IEEE Access, vol. 11, pp. 48322-48331, 2023, doi: 10.1109/ACCESS.2023.3276628
[2] Moghar, A., & Hamiche, M. (2020). Stock market prediction using LSTM recurrent neural network. Procedia computer science, 170, 1168-1173
[3] Fan, X.; Tao, C.; Zhao, J. Advanced Stock Price Prediction with xLSTM-Based Models: Improving Long-Term Forecasting. Preprints 2024, 2024082109.
[4] Bharathi, S., & Geetha, A. (2017). Sentiment analysis for effective stock market prediction. International Journal of Intelligent Engineering and Systems, 10(3), 146-154
[5] Venkataramana, B., Reddy, G. V., Bhaskarareddy, P., & Durga, P. S. (2024). Integrating Sentiment Analysis and LSTM to Predict Price Movements in Stocks. In Disruptive technologies in Computing and Communication Systems (pp. 162-167). CRC Press.
[6] Li, Y., Zhang, X., & Zhu, X. (2023, August). Application of LSTM and Attention Mechanism for Stock Price Prediction and Analysis. In 2023 2nd International Conference on Artificial Intelligence, Internet and Digital Economy (ICAID 2023) (pp. 553-561). Atlantis Press
[7] Mehtab, S., & Sen, J. (2020, November). Stock price prediction using CNN and LSTM-based deep learning models. In 2020 International Conference on Decision Aid Sciences and Application (DASA) (pp. 447-453). IEEE.
[8] Zhou, J. (2024). Predicting Stock Price by Using Attention-Based Hybrid LSTM Model. Asian Journal of Basic Science & Research, 6(2), 145-158
[9] Zhang, J., Ye, L., & Lai, Y. (2023). Stock price prediction using CNN-BiLSTM-attention model. Mathematics 11 (9): 1985.
[10] Nabipour, M., Nayyeri, Pa., Jabani, H., Mosavi, A., & Salwana, E. (2020). Deep learning for stock market prediction. Entropy, 22(8), 840.
[11] Qiu J, Wang B, Zhou C (2020) Forecasting stock prices with long-short term memory neural network based on attention mechanism. PLoS ONE 15(1): e0227222.
https://doi.org/10.1371/journal.pone.0227222.
[12] Pardeshi, K., Gill, S. S., & Abdelmoniem, A. M. (2023). Stock market price prediction: a hybrid LSTM and sequential self-attention based approach. arXiv preprint arXiv:2308.04419.13

[13] Sang, S., & Li, L. (2024). A Novel Variant of LSTM Stock Prediction Method Incorporating Attention Mechanism. Mathematics, 12(7), 945.

[14] Behura, J. P., Pande, S. D., & Ramesh, J. V. N. (2024). Stock Price Prediction using Multi-Layered Sequential LSTM. EAI Endorsed Transactions on Scalable Information Systems, 11(4).

[15] Hargreaves, C. A., & Leran, C. (2020). Stock Prediction Using Deep Learning with Long-Short-Term-Memory Networks. International Journal of Electronic Engineering and Computer Science, 5(3), 22-32.

[16] Mehtab, S., & Sen, J. (2020, November). Stock price prediction using CNN and LSTM-based deep learning models. In 2020 International Conference on Decision Aid Sciences and Application (DASA) (pp. 447-453). IEEE.

[17] Jing, N.W.Z., & Wang, H. (2021). A hybrid model integrating deep learning with investor sentiment analysis for stock price prediction. Expert Systems with Applications, 178: 115019. https://doi.org/10.1016/j.eswa.2021.11 5019.

[18] Aggarwal, P.K., Mittal, R., Ganaie, A., & Ishfaq, M. (2023). Stock prediction by integrating sentiment scores of financial news and MLP- Regressor: A machine learning approach. Procedia Computer Science, 218: 1067–1078. https://doi.org/10.1016/j.procs.2023.01.086.

[19] Sen, J., Mehtab, S., & Dutta, A. (2021). Stock price prediction using machine learning and LSTM-based deeplearning models. https://doi.org/10.36227/techrxiv.15103602.

[20] Wu, J.M.T., Li, Z., Herencsar, N., Vo, B., & Lin, J. (2021). A graph-based CNN-LSTM stock price prediction algorithm with leading indicators. Multimedia Systems, 29. https://doi.org/10.1007/s00530-021-00758-w.

[21] Zhang, Z. (2022). Research on stock price prediction based on PCA-LSTM model. Academic Journal of Business Management, 4(3): 42– 47. https://doi.org/10.25236/AJBM.2022.040308.

[22] Gers, F.A., Schmidhuber, J., & Cummins, F.A. (2000). Learning to forget: Continual prediction with LSTM. Neural Computation, 12(10): 2451–2471.

[23] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Pages 785–794. https://doi.org/ 10.1145/2939672.2939785.

[24] Verma, N. (2022). XGBoost algorithm explained in less than 5 minutes. Medium. https://medium.com/@techynilesh/xgboost-algorithm-explained-in-less-than-5-minutes b561dcc1ccee.